Auddard & Co.

Arithmetic Expression Evaluator in C++
Software Development Plan
Version 1.0

Revision History

Date	Version	Description	Author
<24/9/2023>	<1.0>	Created Project Plan and filled out all necessary information	Audrey Pan, Sriya Annem, Morgan Nguyen, Lauren D'Souza, Ella Nguyen, Tanu Sakaray

Table of Contents

Software Development Plan

1. Introduction

1.1 Purpose

The purpose of this Software Development Plan is to provide a comprehensive strategy and framework for the development of the Arithmetic Expression Evaluator in C++ project. This plan outlines the scope, objectives, roles, responsibilities, and deliverables for the project, ensuring that it results in a functional software feature.

The following people contributed and plan to use this *Software Development Plan*:

- **Project Manager** Audrey Pan uses it to plan the project schedule and resource needs, and to track progress against the schedule.

- **Project Team Members** Tanushri Sakaray, Sriya Annem, Ella Nguyen, Lauren D’Souza, Morgan Nguyen use it to understand what they need to do, when they need to do it, and what other activities they are dependent upon.

1.2 Scope

- This *Software Development Plan* describes the overall plan for developing the Arithmetic Expression Evaluator to be used by the Auddard & Co project, including deployment of the feature.
- This software will be able to handle expressions enclosed within parentheses to determine the correct order of evaluation. The scope also includes the creation of various project management artifacts, including a project plan, a requirements document, a design document, and test cases.
- Additionally, the project entails the development of a user-friendly command-line interface for entering expressions and viewing calculated results.
- Robust error handling mechanisms will be implemented to manage scenarios such as division by zero or invalid expressions.
- Initially, the project assumes input values to be integers. However, the scope allows for potential future changes, such as accommodating floating-point input values, which may be introduced through change requests during project execution.

1.3 Definitions, Acronyms, and Abbreviations

See the Project Glossary in section 6.1.

1.4 References

SN	Title	Published Date	Publishing Organization
1.	EECS348: Term Project in C++ Specification	09/Sep/23	Professor Hossein Saiedian
2.	Expression Parser		Lawrence University
3.	The rules for the order of operations: The case of an inservice teacher	02/Mar/16	HAL Open Science

1.5 Overview

The *Software Development Plan* (SDP) for the Arithmetic Expression Evaluator project in C++ encompasses a clear project overview, including its purpose, scope, objectives, and deliverables. It outlines the project's organizational structure, external interfaces, and states the roles and responsibilities of team members and stakeholders. In the management process section, it details cost estimates, schedules, key phases, and milestones, along with the methods for ongoing project monitoring. The annexes provide insights into the chosen software development process, including methodologies, tools, and techniques, while the appendix contains a Project Glossary for clarifying essential project-related terminology, which serves as a comprehensive guide for the project's successful execution.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The purpose of this project is to develop a C++ program that can parse and evaluate arithmetic expressions, including operators such as +, -, *, /, %, and ^. The program's main goal is to create an arithmetic expression evaluator, that adheres to the order of operations (PEMDAS) and handles parentheses correctly. This project serves as the beginning to becoming familiar with software engineering projects in groups, focusing not only on the final product but also on the development process, including project planning, requirements documentation, design, testing, and error handling.

2.2 Assumptions and Constraints

1. The input expressions will be provided in a valid format, adhering to the rules of math (e.g.no division by zero).
2. The initial input will consist of integer values, but future changes can support floating-point numbers.
3. The project will use object-oriented programming principles to structure the code.
4. The project team has access to the necessary tools and resources to carry out their duties

2.3 Project Deliverables

1. **Project Management Plan**: An initial plan that outlines the project's scope, objectives, schedule, and resource allocation.
2. **Requirements Document** A document specifying the functional and non-functional requirements of the arithmetic expression evaluator.
3. **Design Specs**: A document that outlines the system architecture, data structures, algorithms, and design patterns used in the project.
4. **Test Cases**: A set of test cases taken from the requirements and design to verify if the expression evaluator is correct.
5. **Code**: A well-documented C++ program that implements the arithmetic expression evaluator with support for specified operators and features.

6. **User Manua**l: A user manual or README file explaining how to use the program, including examples.
7. The delivery dates for these artifacts will be specified in the project schedule, which can be found in Section 4.2.4 Project Schedule. These deliverables are identified in the Development Case.

2.4 Evolution of the Software Development Plan

[A table of proposed versions of the ***Software Development Plan****, and the criteria for the unscheduled revision and reissue of this plan. The text below is provided as an example.]*

This table outlines the versions of the Software Development Plan and the criteria for unscheduled revisions and reissues of the plan. The plan is expected to evolve to accommodate changes in project objectives, scope, or resource constraints.

VERSION	***DESCRIPTION OF CHANGES***	***CRITERIA FOR REVISION***
Initial Plan	- creation of the initial plan	N/A first version
Version 1.0	- incorporation of detailed project objectives and scope - clarification of project deliverables - inclusion of the project organization plan - estimation of the projects cost and schedule	- significant changes in the project objective or scope - identification of important resource contains that could impact the plan
Version 1.1	for future use	
Version 2.0		

The Software Development Plan will have revisions mainly to adapt to changing project requirements and conditions. Unscheduled revisions may occur when significant changes or challenges are encountered during the project's execution. The plan will be reviewed and updated at the start of each major project phase to ensure alignment with project goals and objectives.

3. Project Organization

3.1 Organizational Structure

Team Administrator: Audrey Pan

- *Sets up team meetings*
- *Organizes team disagreements*
- *Takes and posts minutes for each team meeting*
- *Manages meetings, including:*
  - *Organizing an agenda for each meeting*
  - *Conducting meeting*

  - o *Working on project deliverables*
- *Brings up any personal issues to the professor*

Product Owner/Utility Developer: Sriya Annem

- *Responsible for user stories*
- *Identifies product features and attributes*
- *Represents the users and customers*
- *Reviews work and help test the product*
- *Works on project deliverables*

Project Lead 1: Morgan Nguyen

- *Compiles and submits all project deliverables*
- *Organizes and splits artifacts between co-project leader*
- *Directs the project and leads the project portion of meetings*
- *Reports any technical issues (that are not resolvable to the team) to the professor*
- *Works on project deliverables*

Project Lead 2: Lauren D'Souza

- *Compiles and submits all project deliverables*
- *Organizes and splits artifacts between co-project leader*
- *Directs the project and leads the project portion of meetings*
- *Reports any technical issues (that are not resolvable to the team) to the professor*
- *Works on project deliverables*

Technical Lead: Ella Nguyen

- *Provides technical guidance and oversight for the project*
- *Ensures adherence to coding standards and best practices*
- *Conducts code reviews and maintain code quality*
- *Identifies and proposes solutions for technical challenges*
- *Collaborates with the team to resolve technical issues and contributes to project deliverables*

Data Administrator/Quality Assurance: Tanu Sakaray

- *Manages data-related aspects of the project, including data quality and security*
- *Develops and implements data management and quality assurance processes*
- *Validates and tests data to ensure integrity and accuracy*
- *Creates and maintains test cases for data quality and software functionality*
- *Performs quality assurance and regression testing, documenting and reporting issues*

3.2 External Interfaces

Person	**Roles/Responsibilities**	**Contact Notes**
Hossein Saiedian	Professor/Instructor	saiedian@ku.edu
Mahitha Bayyapu	Teaching Assistant	mahitha.b@ku.edu

Agraj Magotra	Teaching Assistant	agrajmagotra@ku.edu
Toye Oloko	Teaching Assistant	toye@ku.edu
Liangqin Ren	Teaching Assistant	liangqinren@ku.edu
Nemath Shaik	Teaching Assistant	nyamtulla.shaik@ku.edu

3.3 Roles and Responsibilities

Person	**Unified Process for EDUcation Role**	**Availability Notes**
Audrey Pan	Team Administrator	● ALL AVAILABLE: - Monday night - Tuesday night - Wednesday night - Saturday night ● MOSTLY AVAILABLE: - Sunday afternoon - Sunday evening - Sunday night - Thursday evening - Friday afternoon - Friday evening - Saturday afternoon ● When2Meet
Sriya Annem	Product Owner/Utility Developer	
Morgan Nguyen	Project Lead 1	
Lauren D'Souza	Project Lead 2	
Ella Nguyen	Technical Lead	
Tanu Sakaray	Data Administrator/Quality Assurance	

Anyone on the project can perform Any Role activities.

4. Management Process

4.1 Project Plan

4.1.1 Iteration Objectives

1. Iteration 1
    a. Objective: Complete Project management plan
        i. Lay out the groundwork for this project including roles, deadlines, and other necessary details pertaining to the project
2. Iteration 2
    a. Objective: Requirements Document
        i. Specify the functional and non-functional requirements of the arithmetic expression evaluator in a thorough document.
3. Iteration 3
    a. Objective: Design specs
        i. Outline the system architecture, data structures, algorithms, and design patterns used in the project.
4. Iteration 4
    a. Objective: Project implementation
        i. A well-documented C++ program that implements the arithmetic expression evaluator with support for following operators and features:
            1. Expression parsing - parse arithmetic expressions entered by the user,

taking into account operator precedence and parentheses

2. Operator Support - support for the following operators:
    a. + (addition)
    b. - (subtraction)
    c. * (multiplication)
    d. / (division)
    e. % (modulo)
    f. ^ (exponentiation)
3. Parenthesis Handling - Handle expressions enclosed within parentheses to determine the order of evaluation
4. Numeric Constants - Recognize and calculate numeric constants within the expression

5. Iteration 5
    a. Objective: Test cases
        i. Run and test the project
6. Iteration 6
    a. Objective: User manual
        i. Develop a user manual that explains how to use the program with examples

4.1.2 Releases

Beta v1.0 - projected release 11/19/23

- Our first project implementation of the calculator

4.1.3 Project Schedule

Objective	**Target Completion Date**
Iteration 1	09/24/23
Iteration 2	10/29/23
Iteration 3	11/05/23
Iteration 4	11/19/23
Iteration 5	12/03/23
Iteration 6	12/07/23

4.2 Project Monitoring and Control

4.3 Quality Control

Defects will be recorded and tracked as Change Requests, and defect metrics will be gathered (see Reporting and Measurement below).

All deliverables are required to go through the appropriate review process, as described in the Development Case. The review is required to ensure that each deliverable is of acceptable quality, using guidelines and checklists.

Any defects found during review which are not corrected prior to releasing for integration must be captured as Change Requests so that they are not forgotten.

4.4 Risk Management

Risks will be identified in the Inception Phase using the steps identified in the RUP for Small Projects activity “Identify and Assess Risks”. Project risk is evaluated at least once per iteration and documented in this table.

Refer to the Risk List Document (CCC-DDD-X.Y.doc) for detailed information.

4.5 Configuration Management

Appropriate tools will be selected which provide a database of Change Requests and a controlled versioned repository of project artifacts.

All source code, test scripts, and data files are included in baselines. Documentation related to the source code is also included in the baseline, such as design documentation. All customer deliverable artifacts are included in the final baseline of the iteration, including executables.

The Change Requests are reviewed and approved by one member of the project, the Change Control Manager role.

Refer to the Configuration Management Plan (EEE-FFF-X.Y.doc) for detailed information.

5. Annexes

The project will follow the UPEDU process.

Other applicable process plans are listed in the references section, including Programming Guidelines.

Annex A: Programming Guidelines

1. Coding Standards
    - Use object-oriented programming principles to structure your code
    - Variable and Function Naming: Variables and functions should use meaningful names that reflect their purpose.
    - Indentation and Formatting: Use consistent indentation and follow a consistent code formatting style for readability
    - Comments: include comments and documentation to explain the logic and functionality of your program
2. Code Review Process
    - Code reviews will occur after each major development iteration
    - Technical Lead will conduct these code reviews in a timely manner
    - Review criteria: checking for code correctness, coding standards are enforced
3. Error Handling
    - Use try-catch blocks for handling exceptions ensuring the program’s error recovery
    - Log errors and exceptions with meaningful error messages to aid in debugging
    - Ensure that your program provides clear and informative error messages for invalid input
4. Testing
    - Test code in small units to ensure proper functionality
    - Gradually include portions of code to the whole and check if the program still functions properly

Annex B: Design Guidelines

1. Architecture Overview
   - Use a modular architecture with distinct component for input processing, calculation and output presentation
   - implement a user-friendly command-line interface for interaction
2. User Interface Design
   - Design a clean and intuitive command-line interface with clear prompts and error messages
   - ensure that user inputs are validated before performing calculations
3. Performance
   - Focus on optimizing the calculator for fast and accurate arithmetic calculations
   - implement efficient algorithm for arithmetic operations

Annex B: Process Guidelines (UPEDU)

1. Inception
   - In the initial phase, the Project Leads will focus on understanding the project requirements, including the specific features and functionalities required for the arithmetic calculator
   - The Product Owner will gather user stories and understand user needs and expectation
2. Elaboration
   - The Project Leads will lead the elaboration phase which involves creating a project plan and dividing tasks among team members
   - The team will collaboratively create a project schedule, identifying milestones and deadlines for each phase of development
3. Construction
   - The construction phase involves the actual development of the arithmetic calculator. The Technical Lead will oversee technical aspects and ensure that coding standard are followed
   - Project Leads will manage the construction phase, ensuring that the tasks are completed as per the project plan
4. Transition
   - Once the arithmetic calculator is developed and tested, it will transition to the usage phase, where it will be made available for use by the intended users and stakeholders
   - The Product Owner will represent the users and gather feedback to ensure that the calculator meets user expectations
5. Documentation
   - Throughout all phases, the team will maintain comprehensive project records and documentation, following UPEDU's emphasis on documentation as a key practice
   - The Data Administrator/Quality Assurance role will also document data management strategies, data quality processes, and test cases for data validation

This annex outlines the process guidelines for the arithmetic calculator project following the UPEDU (Understand, Plan, Execute, Document, and Use) process.

6. Appendix

6.1 Glossary

PEMDAS (Order of Operations): Acronym for Parentheses, Exponents, Multiplication, Division (left to right), Addition, and Subtraction (left to right) – the order of arithmetic operations.

Unary Operator: An operator acting on a single operand, e.g., unary minus (-).

Tokenize: The process of splitting an input into individual elements for processing.

Stack: A data structure following Last-In-First-Out (LIFO) for managing function calls and expressions.

Tree: A hierarchical structure used for expression or hierarchical representation.

Operator Precedence: The priority order for evaluating operators.

Numeric Constant: Fixed numerical value in expressions (integers, potentially floating-point).

CLI (Command-Line Interface): Text-based user interaction with software.

Error Handling: Managing unexpected situations, e.g., division by zero or invalid input.

SDP Version: Document version number indicating revisions.

Gantt Chart: Visual project schedule representation.

Change Requests: Formal requests for project requirement, scope, or deliverable changes.

Configuration Management: Controlling changes to project artifacts, versioning, and naming.

UPEDU Process: Unified Process for EDUcation, used for educational purposes.

SN (Serial Number): A serial number is a unique code used to identify and track individual items or products.

SDP (Software Development Plan): A document outlining the software development project's approach, strategies, and key aspects.